دانشکده مهندسی برق

نام دانشجو: کسری خلفی

شماره ی دانشجویی : ۹۵۲۳۰۳۸

استاد درس : دکتر فرزانه عبداللهی

سرپرست ازمایشگاه: مهندس امینی

آزمایش شماره ی پنجم

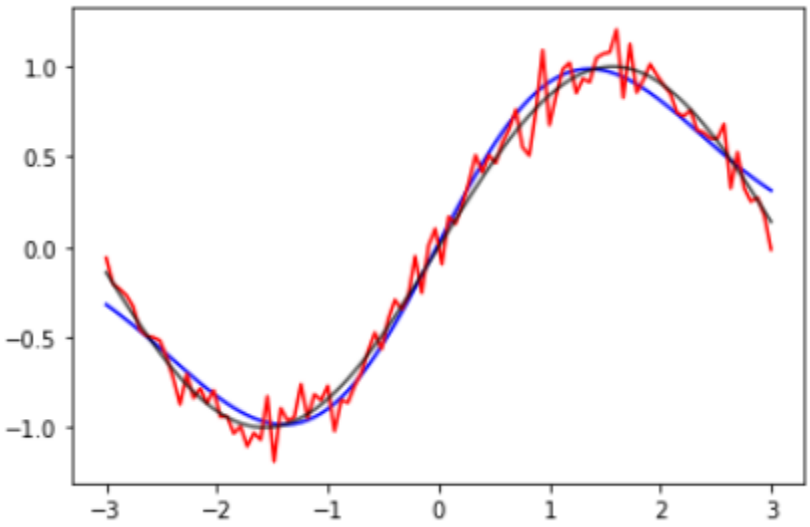
پایتون:

در قسمت پایتون مانند روش یاد شده ۱۰۰ داده ی گسسته در بازه ی -۳ تا ۳ در نظر گرفتیم و به نویز سفید در هم امیختیم.

این روش دارای سه لایه است که ورودی ها به لایه ی اول داده میشود و بسیار شبیه چند لایه ها میباشند با این تفاوت که تابع فعال ساز وسط تابع گاوسین میباشد. تابع را مانند یاد شده تعریف کرده و ۴ دسته مرکز در نظر میگیریم. سپس یک ماتریس به نام ماتریکس تعریف کرده به صورت سطری که سطر اول مقدار یک به عنوان بایاس است و سطر های بعدی خروجی های گاوسین میباشند . سپس این ماتریکس را ترنسپوز کرده تا به صورت ستونی در بیاید.

ضرب های لازم را انجام داده و شکل های به صورت زیر میشود که قرمز ورودی نویزی است و دو خط دیگر خروجی ما و خروجی مطلوب میباشد.

کد ها نیز ضمیمه گشته اند.



F: ▸ university ▸ Term 7 ▸ Computational Intelligence ▸ LAB ▸ Jalase 4 ▸ rbf.py ▸ ...

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans


def gaussian(x, mu, sig):
    return np.exp(-np.power(x - mu, 2.) / (2 * np.power(sig, 2.)))

noron_num = 4

input = np.linspace(-3, 3, 100)
outputDesired = np.sin(input)
outputNoisy = outputDesired + np.random.normal(0, 0.1, input.shape)
# plt.plot(input,outputDesired)
# plt.plot(input,outputNoisy)
# plt.show()

kmeans = KMeans(n_clusters= noron_num,random_state=0).fit(input.reshape(-1, 1))
centers = kmeans.cluster_centers_

max_dist = -1
for i in range(noron_num):
    for j in range(noron_num):
        dist = abs(centers[i] - centers[j])
        if dist > max_dist:
            max_dist = dist

matrix = np.ones((noron_num+1, len(input)))
for i in range(noron_num):
    matrix[i+1] = gaussian(input, centers[i], noron_num / (np.sqrt(2 * max_dist)))
matrix = np.transpose(matrix)
W = np.dot(np.linalg.inv(np.dot(np.transpose(matrix),matrix)),np.transpose(matrix))
W = np.dot(W,outputNoisy)

actual_output = []
for i in range(len(input)):
```

```python
        dist = abs(centers[i] - centers[j])
        if dist > max_dist:
            max_dist = dist

matrix = np.ones((noron_num+1, len(input)))
for i in range(noron_num):
    matrix[i+1] = gaussian(input, centers[i], noron_num / (np.sqrt(2 * max_dist)))
matrix = np.transpose(matrix)
W = np.dot(np.linalg.inv(np.dot(np.transpose(matrix),matrix)),np.transpose(matrix))
W = np.dot(W,outputNoisy)

actual_output = []
for i in range(len(input)):
    actual_output.append(np.dot(matrix[i], W))
plt.plot(input, actual_output, 'b')
plt.plot(input, outputNoisy, 'r')
plt.plot(input, outputDesired, 'black', alpha=0.7)
plt.show()
```

متلب:

مانند کد بالا کد متلب به صورت زیر میباشد:

```
x = 0;
mu = 0;
sig = 0;
y = exp(-power(x - mu, 2.) / (2 * power(sig, 2.)));

noron_num = 4;
input = linspace(-3,3);
outputDesired = sin(input);
%plot(input,outputDesired);
outputNoisy = outputDesired - 0.1 + (0.2)*rand(1,100);
%plot(input,outputNoisy)
%outputNoisy = outputDesired + random.normal(0, 0.1, input.shape)
centers = [-2.3, -0.8, 0.8, 2.3];
max_dist = -1;
for i = 1:noron_num
    for j = 1:noron_num
       dist = abs(centers(:,i) - centers(:,j));
        if(dist > max_dist)
            max_dist = dist;
        end
    end
end

%matrix = ones(noron_num,length(input))
matrix = ones(noron_num+1,length(input));
```

```
for i =1:noron_num
    x = input;
    mu = centers(i);
    sig = noron_num / (sqrt(2 * max_dist));
    matrix(i+1,:) = exp(-power(x - mu, 2.) / (2 * power(sig, 2.)));
end

matrix = transpose(matrix);

W = inv(transpose(matrix)*matrix) * transpose(matrix);
%W = transpose(W)
size(outputNoisy)
size(W)
W = W * outputNoisy;

size(W)
% W = W * outputNoisy
actual_output = zeros(1,length(input));

for i = 1:length(input)
    actual_output(i) = matrix(i) * W(rem(i,4)+1);
end

plot(input, actual_output);
```